CROSS REFERENCE TO RELATED APPLICATIONS

[0001] U.S. 60/450,682

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0002] Not Applicable

REFERENCE TO A MICRO FICHE APPENDIX

[0003] Not Applicable

BACKGROUND OF INVENTION

[0004] 1. Field of Invention

[0005] Generation of entities to create economic value as in corporations , represent the aggregation of complementary skills and visions that create products that are expected to have a value to a set of markets that is greater than the cost of the utilization of all of the constituent participants and the resources brought to bear on the products. In the past these entities were largely created by manual searches for suitable ca~~o~~n~~a~~didates in each functional area and~~dn~~ industry area required for the product by entrepreneurs who used largely informal networks for this process.

[0006] The present invention creates a new paradigm for the creation of economic value generating entities by using Trust networks and Professional Networks. In many embodiments of the invention, these Trust and Professional networks are created by aspects of the invention.

SUMMARY

OBJECTS & ADVANTAGES

[0007] The objects and advantages of the present invention relate to the creation of entities that can generate economic value by utilizing the structure and content of communications among members of information networks using computing means that are created both for social and professional interaction. Such economic value in the present invention may for example relate to the creation of business entities or a new approach for matching job seekers to job offerings with information in normal interactions between members on one or more networks. The invention defines a new approach

Backward chaining

From Wikipedia, the free encyclopedia

Backward chaining (or **backward reasoning**) is an inference method used in automated theorem provers, proof assistants and other artificial intelligence applications. It is one of the two most commonly used methods of reasoning with inference rules and logical implications – the other is forward chaining. Backward chaining is implemented in logic programming by SLD resolution. Both rules are based on the modus ponens inference rule.

Backward chaining starts with a list of goals (or a hypothesis) and works backwards from the consequent to the antecedent to see if there is data available that will support any of these consequents. An inference engine using backward chaining would search the inference rules until it finds one which has a consequent (**Then** clause) that matches a desired goal. If the antecedent (**If** clause) of that rule is not known to be true, then it is added to the list of goals (in order for one's goal to be confirmed one must also provide data that confirms this new rule).

For example, suppose that the goal is to conclude the color of my pet Fritz, given that he croaks and eats flies, and that the rule base contains the following four rules:

1. **If** X croaks and eats flies – **Then** X is a frog
2. **If** X chirps and sings – **Then** X is a canary
3. **If** X is a frog – **Then** X is green
4. **If** X is a canary – **Then** X is yellow

This rule base would be searched and the third and fourth rules would be selected, because their consequents (**Then** Fritz is green, **Then** Fritz is yellow) match the goal (to determine Fritz's color). It is not yet known that Fritz is a frog, so both the antecedents (**If** Fritz is a frog, **If** Fritz is a canary) are added to the goal list. The rule base is again searched and this time the first two rules are selected, because their consequents (**Then** X is a frog, **Then** X is a canary) match the new goals that were just added to the list. The antecedent (**If** Fritz croaks and eats flies) is known to be true and therefore it can be concluded that Fritz is a frog, and not a canary. The goal of determining Fritz's color is now achieved (Fritz is green if he is a frog, and yellow if he is a canary, but he is a frog since he croaks and eats flies; therefore, Fritz is green).

Note that the goals always match the affirmed versions of the consequents of implications (and not the negated versions as in modus tollens) and even then, their antecedents are then considered as the new goals (and not the conclusions as in affirming the consequent) which ultimately must match known facts (usually defined as consequents whose antecedents are always true); thus, the inference rule which is used is modus ponens.

Because the list of goals determines which rules are selected and used, this method is called goal-driven, in contrast to data-driven forward-chaining inference. The backward chaining approach is often employed by expert systems.

Programming languages such as Prolog, Knowledge Machine and ECLiPSe support backward chaining within their inference engines.